Docker Compose Cheatsheet

This is a cheatsheet for docker compose.

Categories: cheatsheet docker-compose

Access this page by simply typing in "**cs docker-compose**" in your browser address bar if you have bunnylol configured.

Commands

Basics

Copy

```
# Build or rebuild services
docker compose build

# Creates containers for a service.
docker compose create

# Start services
docker compose start

# Restart service containers
docker compose restart

# Create and start containers
docker compose up

# Run a one-off command on a service.
docker compose run

# Pause services
docker compose pause

# Unpause services
docker compose unpause

# Stop services
docker compose stop

# Removes stopped service containers
docker compose rm

# Stop and remove containers, networks
docker compose down

# Force stop service containers.
docker compose kill

# Execute a command in a running container.
docker compose exec
```

Check Status

Copy

```
# List running compose projects
docker compose ls

# List containers
docker compose ps

# Print the public port for a port binding.
docker compose port

# Receive real time events from containers.
docker compose events

# View output from containers
docker compose logs

# Display the running processes
docker compose top
```

Images

Copy

```
# Pull service images
docker compose pull

# Push service images
docker compose push

# List images used by the created containers
docker compose images
```

File Operations

Copy

```
# Copy files/folders from a service container to the local filesystem
docker compose cp <service>:<src-filepath> <dst-filepath>

# Copy files/folders from the local filesystem to a service container.
docker compose cp <src-filepath> <service>:<dst-filepath>
```

Others

Copy

```
# Show the Docker Compose version information
docker compose version

# Converts the compose file to platform’s canonical format
docker compose convert
```

Docker Compose Config

An Basic Example

```yaml
version: '3.9'

services:
    hello-world:
        image: hello-world:latest
```

Spec

Build

```yaml
services:
  web:
    # Build from Dockerfile
    build: .

    # Build arguments.
    args:
      APP_HOME: app

    # Build from custom Dockerfile
    build:
      context: ./dir
      dockerfile: Dockerfile.dev

    # Build image.
    image: debian
    image: ubuntu
    image: ubuntu:20.04
```

Network

```yaml
services:
  web:
    # Set container network mode.
    network_mode: "host"
    network_mode: "none"
    network_mode: "service:[service name]"

    # Define the networks that service containers are attached to.
    networks:
      - some-network
      - other-network

    # Expose container ports.
    ports:
      - "3000"
      - "3000-3005"
      - "8000:8000"
      - "9090-9091:8080-8081"
      - "49100:22"
      - "127.0.0.1:8001:8001"
      - "127.0.0.1:5000-5010:5000-5010"
      - "6060:6060/udp"

    # Define dns server.
    dns: 8.8.8.8

    # Define custom DNS search domains to set on container network interface configuration.
    dns_search: example.com

    # List custom DNS options to be passed to the container’s DNS resolver.
    dns_opt:
      - use-vc
      - no-tld-query

    # Defines a network link to containers in another service.
    links:
      - db
      - db:database
      - redis
```

Environment Variable

```yaml
services:
  web:
    # Define environment variables.
    environment:
      RACK_ENV: development
      SHOW: "true"
      USER_INPUT:
      COMPOSE_PROJECT_NAME: "foo"

    # Define environment variables from file.
    env_file: .env
    env_file:
      - ./a.env
      - ./b.env
```

Commands

```yaml
services:
  web:
    # Start up command, which overrides the image default command.
    command: echo "I'm running ${COMPOSE_PROJECT_NAME}"

    # Start up command in the list form, which overrides the image default command.
    entrypoint:
      - php
      - -d
      - zend_extension=/usr/local/lib/php/extensions/no-debug-non-zts-20100525/xdebug.so
      - -d
      - memory_limit=-1
      - vendor/bin/phpunit
```

Labels

```yaml
services:
  web:
    # Container label meta data.
    labels:
      com.example.description: "Accounting webapp"
      com.example.department: "Finance"
      com.example.label-with-empty-value: ""
```

Logging

```yaml
services:
  web:
    # Define logging.
    logging:
      driver: syslog
      options:
        syslog-address: "tcp://192.168.0.42:123"
```

Dependencies

```yaml
services:
  web:
    build: .
    # Define startup and shutdown dependencies between services.
    depends_on:
      - db
      - redis
  redis:
    image: redis
  db:
    image: postgres
```